A DISTRIBUTED SOFTWARE ENVIRONMENT FOR COLLABORATIVE WEB COMPUTING

Antonio Pintus, Raffaella Sanna and Stefano Sanna
CRS4 - Center for Advanced Studies, Research and Development in Sardinia, Italy

Keywords: Distributed, DHT, DART, Web, mobile, collaboration, Web services.

Abstract: This paper describes an extensible core software element of a distributed, peer-to-peer system, which provides several facilities in order to help the implementation of collaborative, Web-based, distributed information storing and retrieval applications based on a decentralized P2P model. Moreover, after an architectural introduction of the core distributed software module, the Core Node, this paper describes a real application, named DART Node, based on it and designed and implemented within the DART (Distributed Agent-based Retrieval Tools) project, which carries out the idea of the design and implementation of a distributed, semantic and collaborative Web search engine, including mobile devices integration use cases.

1 INTRODUCTION

The Internet is evolving in many directions and what is usually called "Web 2.0" summarizes only some of them. While data providers have been decentralized (users superseded traditional publishers), infrastructure is still centralized, held and controlled by a few companies.

Managing large amounts of data and supporting collaborative participation at infrastructure level, are two of the key concepts on which are been focused the studies and investigations conducted during the DART research project (Distributed Agent-based Retrieval Tools, http://www.dart-project.org) (Angioni et al., 2007).

The main goal of DART is to realize a flexible, P2P, collaborative, scalable, fault-tolerant and self-organized system, which achieves a collaborative storage and retrieval of large volumes of resources, for the implementation of a distributed, semantic and collaborative search engine prototype.

This paper presents and describes the main software components of this P2P distributed system.

2 THE CORE NODE GENERAL ARCHITECTURE

The DART system can be viewed as a federation of nodes called Core Nodes, whose modular architecture is described in this section.

2.1 DHT Layer and DHT Abstraction Layer (DAL)

Distributed Hash Tables (DHT) are considered state-of-art approach to massively distributed and storage-oriented systems (Balakrishnan, 2003). By means of DHTs it's possible to realize networks of cooperating nodes with a deterministic resource localization and an efficient requests' routing.

The Core Node is mainly a DHT node, based on and extending the PAST framework (Peer-to-Peer Archival Storage, http://freepastry.org) (Druschel, 2001), a large-scale persistent and global storage system based on the Pastry routing algorithm (Rowstron et al., 2001), so it basically supports data *insert* and *lookup* operations. Moreover, being a Pastry node, it is also able to route messages with a generic payload.

The DHT Abstraction Layer of the Core Node is a Java API layer (Figure 1) that wraps all the low-level DHT network operations and API provided by PAST and FreePastry frameworks. It exposes an interface which simplifies DHT operations and

Pintus A., Sanna R. and Sanna S. (2008).
A DISTRIBUTED SOFTWARE ENVIRONMENT FOR COLLABORATIVE WEB COMPUTING.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 263-266
DOI: 10.5220/0001720902630266

message sending over the network. All the higher level layers in the Core Node architecture rely on this API.

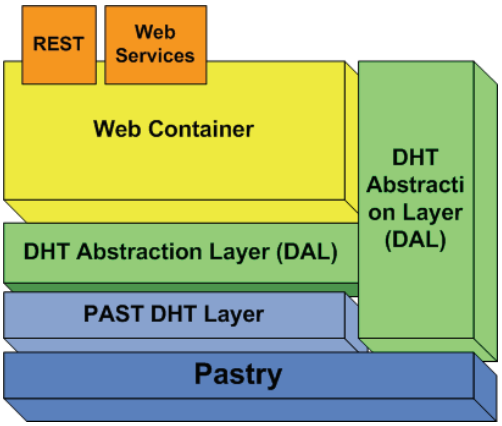

Figure 1: General Architecture of the Core Node.

2.2 Web Container

The Core Node is equipped with a full configurable embedded Java Servlet Container, which allows to fulfil HTTP requests for both static and dynamic contents and applications, like Web Services. This way, it is possible to create and deploy Web applications in the node.

2.3 Services

The node API (Section 2.1) can be exploited in order to design and implement more domain-focused distributed software applications, as in the DART scenario (Section 3). Anyway, in order to grant the interoperability between the node and external heterogeneous software systems, a services layer, or another standard mechanism, become necessary.

Two types of services are provided in the node: RESTful Services and Web Services (SOAP), which wrap and use the underlying DHT Abstraction Layer.

2.3.1 REST Services and Web Services

The Node provides RESTful Services, which expose an interface for all the basic system functionalities, like data storing and retrieval requests, or messages sending.

REST interface is fundamental to allows an access to DART network to mobile devices and embedded systems. Moreover, REST interface simplifies the design and implementation of RIA using AJAX and standard web browsers.

Although REST can be successfully adopted for fast integration of simple components, it is not suitable for complex architectures. REST lacks in formal descriptions of services interfaces and in embedded security management. So, the Core Node, also provides a more formal interface, using standards like WSDL, SOAP and XML Schema, for exposed services which are equivalent to RESTful services mentioned in the previous section.

3 A SEARCH ENGINE APPLICATION: THE DART NODE

The DART research project is focused on studying, developing and testing patterns and integrated tools to improve the quality of search engines results with the main objective to satisfy user needs. Among the others, interesting research fields such semantic-based indexing, P2P crawling, public Web resources indexing, location-aware information retrieval and virtual assistance, are exploited and merged (Angioni et al., 2007).

Studies and investigations conducted in DART, have led to the development of a software application prototype: the DART Node.

3.1 The DART Node Architecture

The DART Node is based on the Core Node, inheriting all the basic functionalities and extending them. At run-time, the DART Node automatically discovers other nodes and collaborates with them in the P2P network, performing Web crawling tasks and storing a portion of the content crawled by all the nodes.

Semantic issues are faced by a Semantic Module (Figure 2), which works on crawled data and performs a semantic and geographical categorization. To achieve this goal, a semantic analysis process on structured and unstructured parts of documents is performed (Angioni et al., 2007).

3.2 Collaborative Crawling System

The collaboration between nodes in crawling activities, helps to crawl the Web in a more effective manner, reducing network traffic and avoiding duplication of tasks and nodes overload.

The DART Node adopts a simple policy for distributed crawling, called “partition by URL”, where the partitioning scheme is determined by the

way URLs are published into the DHT, hashing the entire URL. Each node is responsible for crawling the URLs published in its partition of the DHT. (Loo et al., 2004). Crawling distribution is achieved through a special messages exchange between nodes.

Future work in DART collaborative crawling system may consider the adoption of a topic oriented collaborative crawling (Chung et al., 2002).

3.3 Indexer Module

This module (Figure 2) is capable to collect data coming from (potentially) several data providers, for example from Web crawlers, and to store them in the DHT through the DHT Abstraction Layer.

The Indexer works using a queue with the adoption of a producer-consumer paradigm. For textual data types, the Indexer can use the Semantic Module in order to perform a classification of data before the storing step.

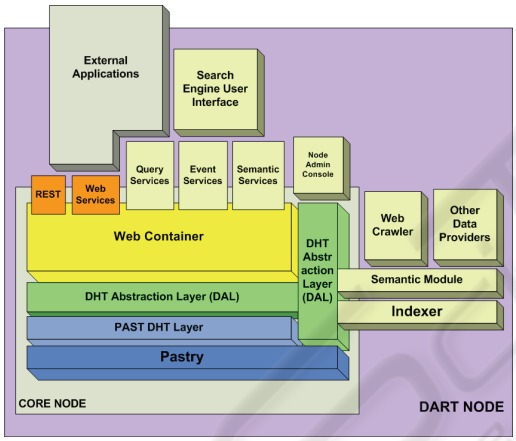

Figure 2: General Architecture of the DART Node.

3.4 DART Node Services

The DART Node exposes RESTful and Web Services providing the following macro-functionalities to potential remote clients:

- Query Services: provide an interface for full life-cycle management of queries submitted to the search engine;
- Semantic Services: by means of the Semantic Module they provide operations to perform semantic classifications of textual resources;
- Event Services: provide access to functionalities related to the DART event delivering and notification system, still under development (see Section 4.1).

Services are implemented using JAX-WS 2.1 and Java Servlet frameworks.

3.5 Mobile Devices Applications

The network of DART nodes is able to store and asynchronously retrieve any kind of data, taking advantage of systematic and redundant distribution provided by DHT. RESTful services have been designed to be accessible through mobile phones and embedded systems. These devices have two key roles: data provisioner and data provider. As data provisioner, a mobile device performs queries on the DART network, to retrieve data and display it to the user. As data provider, a device collects data using sensors and readers and stores such information to be afterwards searched and retrieved by data provisioner. Consumer appliances, like cellular phones, act as provisioner; same devices and embedded systems act as data providers (the latter are intended to collect and automatically publish data through the DART Node without human actions).

Although mobile browsers have been enhanced to access seamlessly standard web sites, they are not suitable to perform asynchronous background operations, access local peripherals and storage. Asynchronous access to services is crucial to improve user experience and to avoid continuous network operations over expensive cellular networks. At the same time, the ability to read data coming from sensors and surrounding appliances (such as RFID readers, GPS and accelerometers) is mandatory to implement mobile data providers.

Mobile DART Node is a stand alone application for Java ME enabled mobile phones that connects to one or more DART Nodes, submits queries, checks for results and retries them asynchronously, basically adopting a pull mechanism.

Mobile DART Node does not replace embedded browser: it runs as a bridge between the DART network and the browser, caching and sorting results, performing auto-updates on queries. Once results have been collected, it provides a summary of them: when the user selects a result, its URL is passed to the web browser for rendering.

Mobile phones equipped with RFID reader and GPS can run Mobile DART Node Data Provider (DP) variant, which allows to publish data through the DART Node REST interface. The Mobile DART Node DP populates the DART network with association about objects (identified by radio tags) and places.

4 WORK IN PROGRESS

4.1 Distributed Event Delivery System

The Core Node, thanks to its architecture can be used in a profitable way to build a distributed, collaborative and failure-resistant Event Delivery System. At the moment, a so described system, is under design and development, adopting a publish/subscribe model and involving mobile sensor-equipped devices.

4.2 The Node in a Service Oriented Architecture (SOA) Context

The service layer of the Core Node, in particular the exposed Web Services, points out the chance of an inclusion of the Node and its derived applications in a SOA context.
Moreover, the distributed system itself can also set up a redundant Web Service Registry (also including semantic issues) which can be used for service publication aims.

5 CONCLUSIONS

The Internet has evolved to a collaborative basis, where information is collected from multiple sources and assembled by users. Collaboration at infrastructure level is still to come. The DART Project aims to propose and realize a flexible, distributed, collaborative, scalable, fault tolerant and self-organized system for a semantic search engine. The proposed software architecture realizes the abstraction layer to DHT framework and exposes storage and retrieval functionalities through SOAP and REST web services. DART network is suitable for both text documents, multimedia content and environmental data coming from distributed sensors. Mobile integration interfaces are core parts of basic architecture and extended prototypes are being developed and tested in real environment.

ACKNOWLEDGEMENTS

The architecture and the prototypes described in this paper belongs to the DART - Distributed Agent-based Retrieval Tools Project at CRS4, partially funded by the Italian Ministry of University and Scientific Research (Contract grant n. 11582).

REFERENCES

Angioni, M., Demontis, R., Deriu, M., De Vita, E., Lai, C., Marcialis, I., Paddeu, G., Pintus, A., Piras, A., Sanna, R., Soro, A., Tuveri, F., 2007. A Collaborative, Semantic and Context-Aware Search Engine. In Proc. of ICEIS 2007 – 9th International Conference on Enterprise Information Systems.

Angioni, M., Demontis, R., Deriu, M., De Vita, E., Lai, C., Marcialis, I., Pintus, A., Piras, A., Soro, A., Tuveri, F., 2007. DART: The Distributed Agent-Based Retrieval Toolkit. In Proc. of 2007 WSEAS International Conference on Computer Engineering and Applications (CEA07).

Angioni, M., Demontis, R., Deriu, M., De Vita, E., Lai, C., Marcialis, I., Pintus, A., Piras, A., Soro, A., Tuveri, F., 2007. User Oriented Information Retrieval in a Collaborative and Context Aware Search Engine. In WSEAS Transactions on Computer Research Journal.

Rowstron, A., Druschel, P., 2001. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proc. of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware).

Druschel, P., Rowstron, A., 2001. PAST: A large-scale, persistent peer-to-peer storage utility. In Proc. of The 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII).

Angioni, M., Demontis, R., Tuveri, F., 2007. Enriching WordNet to Index and Retrieve Semantic Information. In Proc. of 2nd International Conference on Metadata and Semantics Research.

Loo, B. T., Cooper, O., Krishnamurthy, S., 2004. Distributed Web Crawling over DHTs. In UCB/CSD-04-1305, EECS Department, University of California, Berkeley.

Chung, C., Clarke, C. L. A., 2002. Topic Oriented Collaborative Crawling. In Proc. of CIKM'02, Conference on Information and Knowledge Management.

Balakrishnan, H., Kaashoek, M.F., Karger, D., Morris, R., Stoica, I., 2003. Looking up data in P2P systems. In Communications of the ACM, February 2003.